The De Morgan Gazette 5 no. 2 (2014), 5–18 ISSN 2053–1451

ROBOTBASIC IN THE CLASSROOM

JOHN BLANKENSHIP

Introduction

Nearly a third of all students drop out of school[†] because they are not finding education relevant, challenging, or exciting. The changing nature of work, technology, and competition in the global job market often outpaces what our education system provides for students[‡]. The future will require a new workforce of problem solvers, innovators, and inventors who are self-reliant and able to think logically[§]. A key to developing these skills is strengthening science, technology, engineering, and math (STEM) competencies in every K-12 student[¶].

The top 10 "in-demand" jobs for 2010 did not even exist in 2004. More than 1/3 of top 60 future jobs will DIRECTLY require programming, robotics, or engineering skills[‖]. Many of the remaining jobs will enjoy indirect benefits from the skills acquired from these disciplines. Seven of the top skills that will be needed in the future[**] are:

- Critical thinking and problem-solving
- Collaboration across networks and leading by influence
- Agility and adaptability
- Initiative and entrepreneurialism
- Effective oral and written communication
- Accessing and analyzing information
- Curiosity and imagination

Programming and Robotics

Programming robots has been shown to not only motivate students, but improved many of the skills mentioned above. Even the military at West Point has used a Lego robot and a robot simulation to teach such skills to freshmen[††], but with an

---

2010 *Mathematics Subject Classification* 97A80, 97B70.

© 2014 John Blankenship

[†] http://www.boostup.org/en/facts/statistics#fl.

[‡] http://www.leadered.com/pdf/Academic_Excellence.pdf.

[§] http://www.redstickrobotics.org/the-importance-of-stem-learning.html.

[¶] http://www.redstickrobotics.org/the-importance-of-stem-learning.html.

[‖] http://www.getdegrees.com/careers/tips/top-60-jobs-that-will-rock-the-future/.

[**] http://asiasociety.org/education/resources-schools/professional-learning/seven-skills-students-need-their-future.

[††] http://www.robotdiy.com/Downloads/articles/Using+robots+and+simulation+to+teach+problem+solving+in+an+introductory+course+in+computing+and+information+technology.pdf.

appropriate programming language, there is no reason to wait so long to introduce students to programming and robotics.

Students at the 5th grade level have been taught to program a robot using RobotBASIC. The excitement generated by such a class cannot be imagined until it is seen in action. Understanding how to generate this excitement is very important.

Programming a robot can be a challenging task, even for college students, but with an appropriate language, outcomes and expectations can be easily adjusted for a wide variety of grade levels. Students quickly recognize the relevance of the problems to be solved because they get immediate feedback showing the *need* for new knowledge[†]. Furthermore, the constant hands-on activities associated with robotics keeps students engaged in active learning processes. In addition to these direct benefits, there are many indirect ones.

When robots are used in the classroom, students can work in small groups of 2 to 4 students to brainstorm ideas. This encourages the development of basic communication and inter-personal skills[‡] such as the ability to collaborate and convey complex ideas to a fellow students or colleagues. The use of robots in the classroom also subconsciously introduces students to possible career paths they may well have never considered[§].

Finally, programming can be a valuable learning tool in its own right. Any good teacher will tell you that the teaching process itself imparts, to the teacher, a greater understanding of the material being taught. When a student writes a program, they are essentially teaching the computer how to solve the problem. In order to teach the computer (write a program) students must investigate the problem, organize the material, identify solutions, implement them in code, evaluate their effectiveness, and improve upon them by recycling thorough this process[¶].

The advantages of including programming and robotics in a curriculum are summarized below.

- Motivational projects
- Immediate feedback
- Active learning
- Applicable over all grade levels
- Improves communication and interpersonal skills
- Increases awareness of career possibilities
- Teaching the computer improves personal understanding

RobotBASIC

The Robot Simulator extension of RobotBASIC was originally programmed by college Professor John Blankenship. The simulator was incorporated as a set of

[†]http://www.connectivism.ca/?p=16.

[‡]http://tribotix.com/EducationInfo/WhyRobotics.htm.

[§]http://tribotix.com/EducationInfo/WhyRobotics.htm.

[¶]http://www.teachervision.fen.com/problem-solving/teaching-methods/48451.html.

functions and commands into a powerful interpreted programming language written by Samuel Mishal. The combination is a versatile language that is easy to learn and fun to use, even by those new to programming. RobotBASIC is not, however, just a beginner's language. With over 800 powerful commands and functions, experienced programmers can use RobotBASIC to handle complex problems.

The RobotBASIC programming language has features that makes it easy to add programming and robotics to almost any curriculum. RobotBASIC has been designed to be especially effective when implemented in a multi-year program. Legacy-style commands allow even 5^{th} graders to quickly grasp programming concepts. As student's skills and maturity increase, more complex commands can be introduced so more complicated problems can be explored. By high school, students can be introduced to the C-style syntax used in many modern languages. This progression allows students to grow without having to constantly relearn new programming environments while maintaining continuity between classes. Additionally, it prepares students for the cryptic syntax associated with the languages like C and Java that are often used in college classes.

When a comprehensive approach is utilized throughout an entire curriculum, teachers of many disciplines, and at every grade level, can offer programming assignments that support standard coursework. When implemented properly, there is no need to add separate programming courses or significantly alter current coursework. Instead, programming is used to augment what is already being done. This allows the computer to be viewed as a powerful tool (as it is in industry) instead of a subject in and of itself. RobotBASIC's easy-to-use graphics and powerful functions make it easy to create exciting and enlightening assignments for classes involving math, science, and physics. High school students can, for example, program a simulation of gravity (Figure 1) or the physics of a spring (Figure 2) to promote a more complete understanding of the associated principles.

The RobotBASIC Simulator

RobotBASIC's integrated simulator allows robotic programming to be explored at nearly any grade level, without the need for costly hardware—although hardware can be easily added at any time. The simulator's extensive sensory abilities can be very motivational because they allow a wide-variety of meaningful projects to be explored. RobotBASIC's willingness to work with teachers to create customized assignments along with teacher training and support materials can make adoption and integration trouble-free for both faculty and staff.

RobotBASIC's simulated robot has far more capabilities than most educational robots, allowing RobotBASIC to handle projects that are more complex and far more stimulating for students. The sensors available on the simulator are:

- Ultrasonic rangers (rRange) to measure the distance to objects on the screen.
- Infrared detectors (rFeel) to determine if any objects are close to the robot.
- Bumper switches (rBumper) to determine if the robot bumps into an object.
- A Compass (rCompass) to measure the angular orientation of the robot.

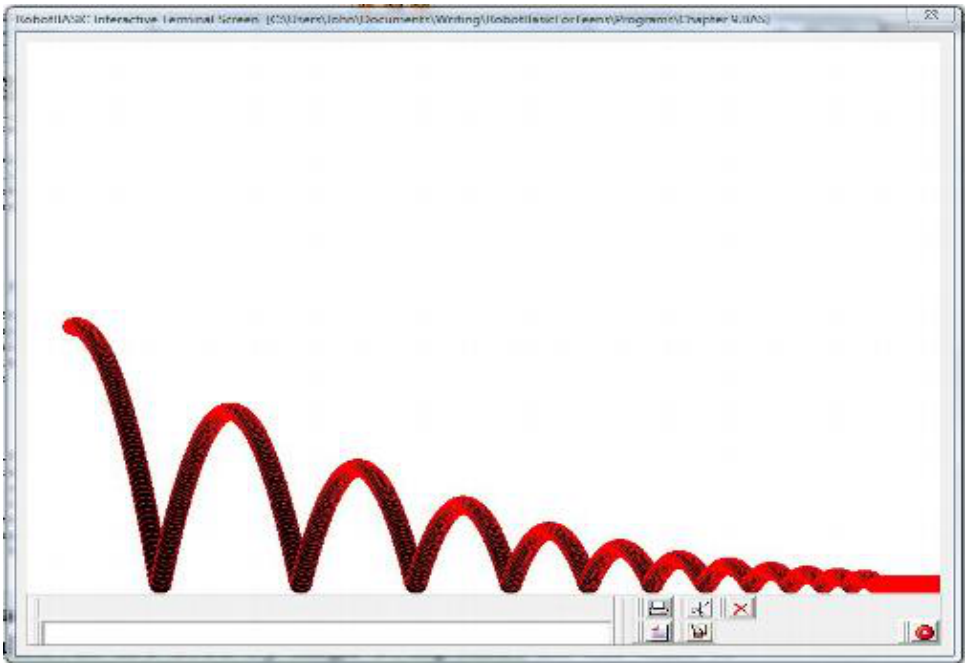

FIGURE 1. *Simulation of gravity*

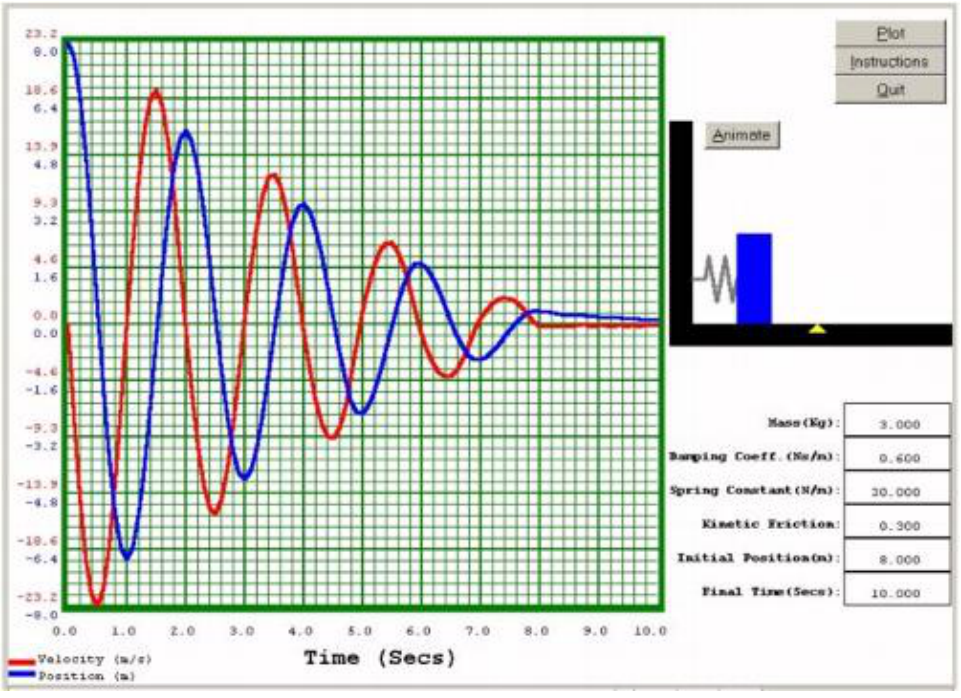

FIGURE 2. *The physics of a spring.*

- A GPS (rGPS) to find out the current X, Y coordinates of the robot.
- Line sensors (rSense) to detect the presence of a line drawn on the floor.
- A Beacon sensor (rBeacon) to determine if the robot is facing a beacon.
- A Camera (rLook) to determine the color seen in front of the robot

With all of these sensors, the robot can be programmed to follow a line (Figure 3), solve a maze (Figure 4), hug a wall (Figure 5), or find its way through a cluttered

environment to connect with a battery charger (Figure 6). These are only a few of the many projects that can be handled with the RobotBASIC simulator. Robotic projects are great teaching tools because students must brainstorm ideas in order to solve exciting and meaningful problems. They get immediate feedback and have to refine their methodology to achieve their goal.

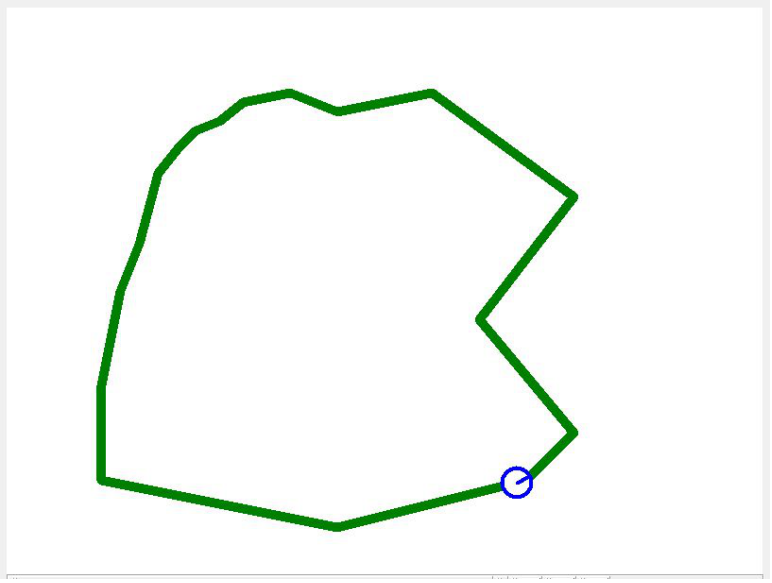

FIGURE 3. *The Robot uses sensors to follow a line on the floor.*

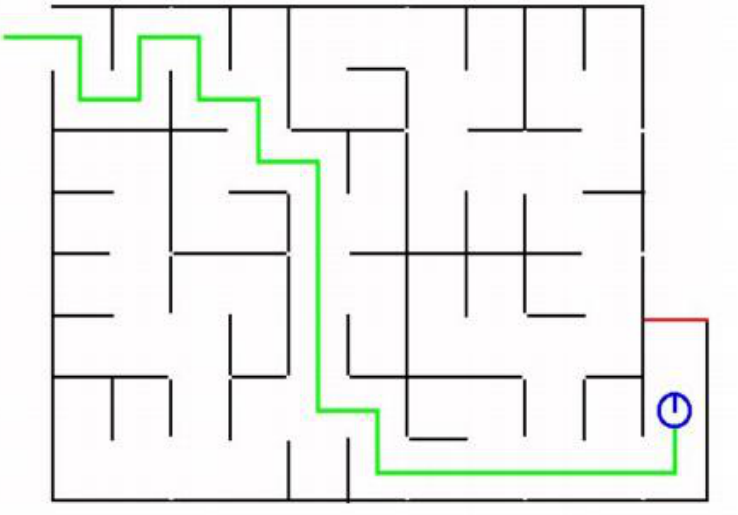

FIGURE 4. *After exploring the maze, a well programmed robot can move directly to the end.*

A summary of the educational advantages of RobotBASIC is given below.

- Easy to use, yet powerful enough to create motivational assignments.
- Multiple versions of basic commands allow students to grow over time without the need to relearn new languages or environments.

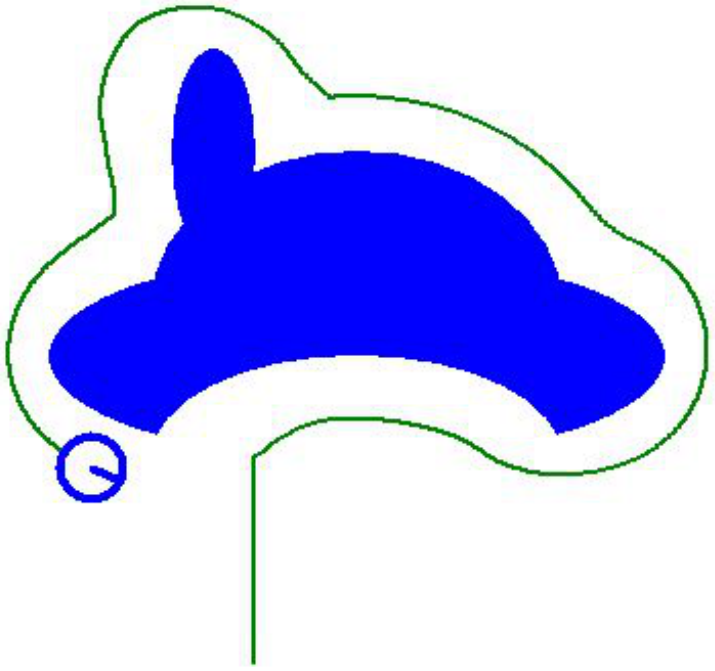

FIGURE 5. *The robot can use its perimeter sensors to "feel" its way around a wall.*

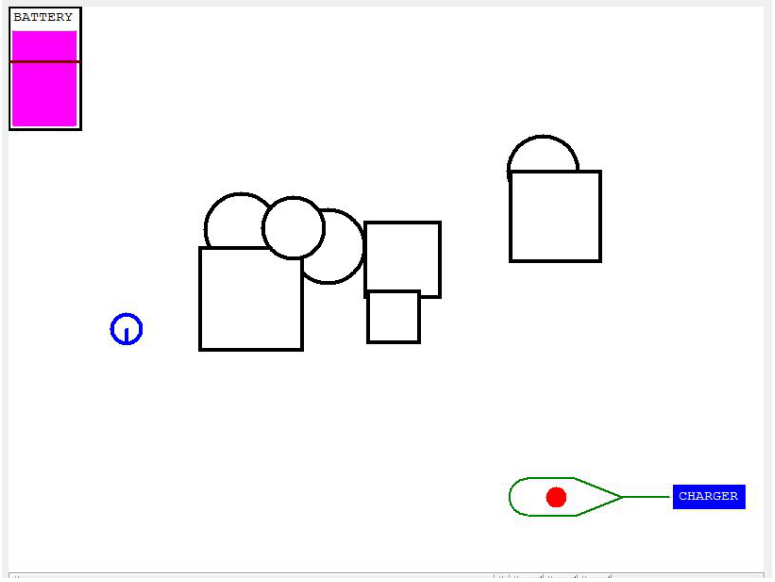

FIGURE 6. *The robot finds its way through a cluttered environment to connect with a charger to replenish its battery.*

- Since all students are exposed to programming, it can be utilized to directly support current curricula. This minimizes the training required for each teacher.
- Optional C-style syntax helps prepare students for college-level programming courses.
- An integrated robot simulator motivates students by making exciting, relevant projects possible at any grade level.

Controlling Real Robots

Real robots with capabilities similar to those of the simulation can also be controlled from RobotBASIC over a wireless Bluetooth link. When a school desires to include real robots in their programs it can be very cost effective because often only one robot per school is needed. This is possible because programs developed for the simulator can immediately be used to control a real robot that is powered by the RobotBASIC Robot Operating System (RROS).

Students can work in small groups to develop ideas for accomplishing assigned tasks, then each student can implement their own solution using their own simulated robot. Once a student gets his or her simulation operating correctly, the teacher simply plugs a USB Bluetooth transceiver into their computer to allow their program to control the real robot.

RobotBASIC provides complete technical information for those that wish to design and build their own RROS-based robots. The easiest way though, is to use the RROS chip and PCB shown in Figure 7. The system supports a wide variety of sensors and motors allowing robots to be built that range from simple to sophisticated.

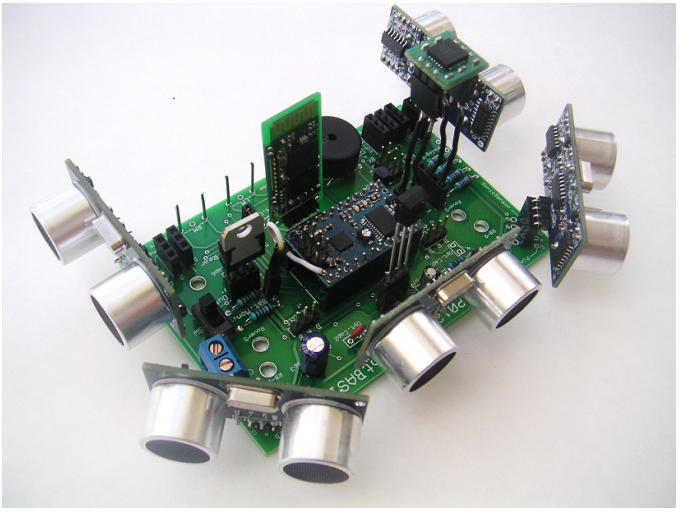

FIGURE 7. *The RROS chip and PCB makes it easy to build a robot regardless of the types of sensors and motors used.*

Figure 8 shows some of the robots that were used to test the RROS while it was being developed. They range in size from less than 8 inches in diameter to a large man-sized robot and are powered by several types of motors and utilize a wide variety of sensors. The RROS PCB can also be mounted to a variety of commercially available chassis as shown in Figures 9, 10, and 11.

Even with all of these capabilities, RobotBASIC is *free*. Schools have *no* acquisition costs, *no* upgrade fees, *no* site licenses—*ever*! And, teachers and students can download their own personal copy of RobotBASIC for preparing assignments and doing homework. RobotBASIC comes with a 300 page HELP file and a huge assortment of demo programs. There are also dozens of YouTube videos to help

FIGURE 8. *Many robots were built and tested while the RROS was being developed.*

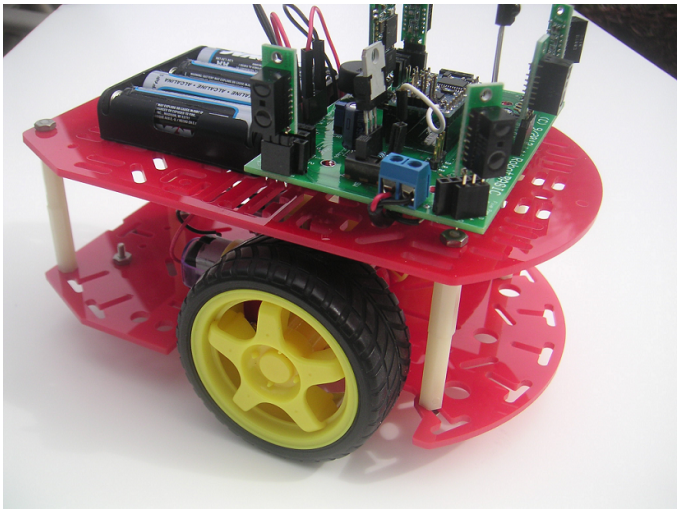

FIGURE 9. *The RROS PCB mounted on a Magician Chassis.*

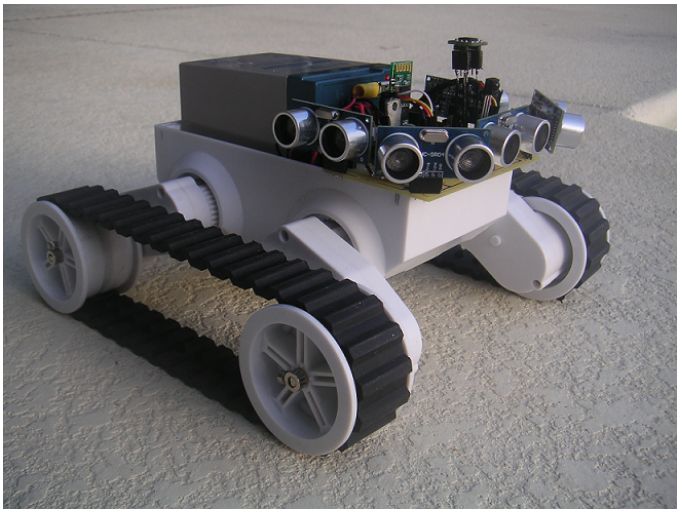

FIGURE 10. *The Rover 5 Chassis is ideal for outdoor use.*

beginners get started. For those that need more formal instruction, there are many books available (see Figure 12).

A Comprehensive Curriculum

We recommend that RobotBASIC be introduced to students in the 5th grade. Assignments given in the early grades are designed to be exciting and motivational, while introducing fundamental programming concepts and ensuring basic competency with the RobotBASIC environment.

As students move to middle school, assignments can introduce new programming principles as well as becoming more mathematically oriented. This allows students to improve their understanding of many topics including Cartesian coordinates, the graphing of equations, working with angles, and other algebraic principles. In many cases, the projects used can directly support current curricula. More complex programming topics should be introduced at this time.

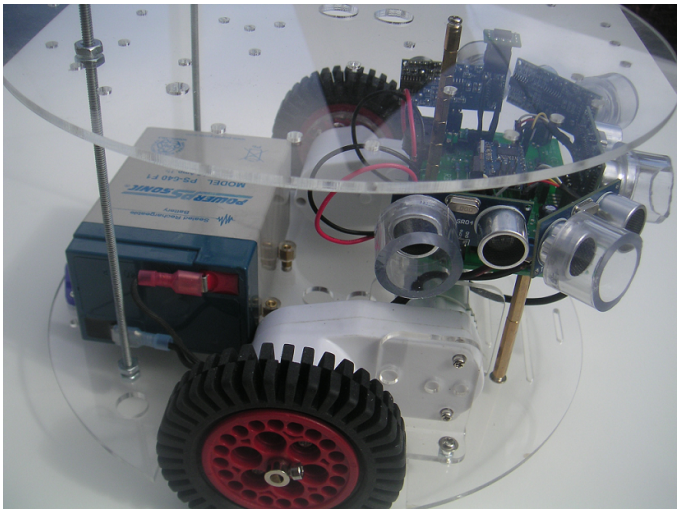

FIGURE 11. *The RobotBASIC RB-9 Chassis is currently under development.*

FIGURE 12. *Numerous books are available to provided more formal help, when needed.* ***RobotBASIC is FREE***

At the high school level assignments can be used in courses involving physics, geometry, trigonometry, and science. Again, the projects should augment, rather than replace existing curricula. More advanced programming topics, including alternate syntax can be introduced at this level.

Highlights of RobotBASIC

RobotBASIC has many features. Here is a summary of some of them.

- No installation is required—you can run RobotBASIC from a USB-drive, a CD,

or even from a web page. This means schools have no networking problems when installing RobotBASIC.

- Use variable typing (integer, float, and string) *or* totally untyped variables, where *any* variable at *any* time can be changed to *any* type by simply assigning it a new value. This feature makes RobotBASIC especially easy for beginners.
- Standard GOSUB routines or callable function-like routines (Call/Sub) with LOCAL variable scoping with by reference and by value parameters.
- Use legacy INPUT and PRINT statements for *quick* and *easy* I/O (great for introducing programming to non-programmers). Advanced students can use GUI commands to create buttons, text boxes, edit boxes, list boxes, dialogue windows, message boxes, radio buttons, check boxes, sliders, and more.
- Use a Procedural programming model with GOSUB and CALL/SUB or an EVENT-DRIVEN approach with commands like OnSlider. Use standard BASIC syntax or a modified C-style syntax (i.e. ++, +=, !=, &&) This can be great for teaching programming fundamentals to beginners before moving on to more complex principles and syntax.
- Increased productivity from numerous "helper functions" that facilitate sorting, multimedia displays, flicker-free 2-D and 3-D animation, robot vision (including web cam support), extensive BMP image manipulation, matrix math, both high and low-level file I/O, the ability to send emails (SMTP) and communicate over the Internet (UDP and TCP protocols), and much more.
- Develop and debug programs in an easy-to-use INTERPRETER-based IDE (Integrated Development Environment), then COMPILE your programs to standalone EXE's for easy distribution.
- Ability to create includable LIBRARY files with ♯include (even include binary files to protect your algorithms). This makes it easy for teachers to provide startup materials for student assignments.
- An integrated Robot Simulator with numerous sensors such as an electronic compass, ultrasonic distance measurement, IR perimeter sensing, line sensors, a GPS, and more.
- Control real robots using parallel, serial, and USB ports for wireless protocols such as Bluetooth and Zigbee. Unique proprietary protocol allows simulator programs (without modification) to control real robots (see our books and the HELP file for complete details).
- Direct support for the USBmicro U4x1 family of I/O modules that provide 1-wire, SPI, and I2C Serial control of Digital MicroDevices as well as 16 lines of TTL I/O.
- Ability to *spawn* other executable files and pass them parameters.
- Over 800 powerful commands and functions often allow a few lines of code to provide the functionality of hundreds of lines in many other languages.
- An extensive 300 page HELP file provides detailed information and numerous programming examples.
- Web page tutorials, sample programs, YouTube videos, and low-cost books provide additional help if needed. Improved editor with multiple file capability and color-coded keywords.
- RobotBASIC is *free* to schools, organizations, individuals—*everyone*!

The RobotBASIC Robot Operating System

The RobotBASIC simulator's sensory capabilities allow it to be used in a wide variety of projects. Once programs are working on the simulator though, it is highly motivational for them to be used to control a real robot that has capabilities similar to that of the simulator. In the past, building such a robot has been a daunting task reserved only for those with significant knowledge and skill of both electronics and low-level microcontroller programming.

Now schools can have a complete hardware/software solution in the form of a 24-pin IC preprogrammed with the RobotBASIC Robot Operating System (RROS).

The RROS not only provides the physical interface needed for many motors and sensors, it also provides the software required to seamlessly interface with all supported devices using the high-level RobotBASIC simulator-based commands and functions. This means that the RROS not only makes it easier than ever before to build a robot, it makes it easier to program one too.

Imagine being able to simply connect an electronic compass to the RROS chip and then immediately determine a real robot's orientation using the same commands used with the simulator—all without ANY low-level programming. Imagine connecting infrared or ultrasonic sensors directly to the RROS chip to create a Virtual Sensor System capable of determining where obstacles are that might block your robots path.

Controlling a robot's motors is just as easy as acquiring its sensory information. When simulator commands like rForward and rTurn are used, the robot won't just respond, it will respond intelligently, automatically ramping up and down when speeds are changed, using compass readings and wheel encoder counts when possible and appropriate.

These examples don't begin to describe the power of the RROS. Download the complete *RROS User's Manual* from `RobotBASIC.org` and you will discover the real power... of simplicity. If you do not have time to read the entire manual (around 250 pages) below is a summary of what to expect.

Highlights of the RROS

- Use a radio link (such as Bluetooth) or a direct serial link from a PC (if your robot uses an embedded PC on board) for communication with RobotBASIC.
- The chip will directly drive DC motors up to a max of 1 amp each or continuous rotation servo motors without external support. For bigger motors, the chip will interface with and control a RoboClaw board allowing control of large (up to 30 amp) motors. The system will automatically slowly increase and decrease the robot's speed as it starts and stops to prevent jerky motion. As with other aspects of the system all controlling parameters can be set from RobotBASIC using the rCommand() function.
- The wheel counters (when present) automatically allow the robot to move spec-

ified distances and rotations more accurately than an open-loop timed response. If wheel counters are not present, there are commands that allow the user to fine-tune the timing loops used to control movements. The timing approach works far better than you might expect, as long as most of the Robot's movements are adjusted based on other sensory data. Full control of your robot's movements is provided through simulator commands such as rForward and rTurn—all the details are handled by the RROS. Additional commands allow you to fine-tune and modify the movements to increase your robot's capabilities.

- Perimeter sensors are translated into appropriate rBumper() and rFeel() signals. You may use a wide variety of sensors including, digital IR devices such as Pololu's Sharp GP2Y0D810 or GP2Y0D805. A more advanced technique is to use analog style sensors so that a VIRTUAL sensory system can be *automatically* created for you. The VIRTUAL system utilizes perimeter ranges sensors to create the data for rRange(), rFeel(), and rBumber(). You can use Ranging IR sensors such as Pololu's Sharp GP2Y0A21 or GP2Y0A02, or ultrasonic rangers such as the Maxbotix Sonar Sensors or even Parallax's PING sensors. Special setup commands allow you to tell the RROS what motors and sensors you are using. In all cases, the data will be collected and translated for use through the RobotBASIC simulator commands. All of the above sensors are supported even beyond typical expectations. For example, when six PING sensors are used for perimeter sensing, they can be read simultaneously using a special algorithm—providing superior sensory performance. Great care was taken to support a wide variety of sensors from various manufacturers—ideally you can build a robot *your* way and yet all the details are done for you.
- Three digital line sensors are supported by the main IC chip. Digital in this case simply means the sensor supplies a logical 1 or 0 as its output. This is true for the Pololu line sensors that are called "analog" sensors. Their "digital" line sensors use a digital port pin, but actually create an analog result based on the time for a capacitor to discharge, and are NOT compatible.
- A beacon detector such as Pololu's Vishay TSOP34156 can be used to detect 15 different beacons as described in some of our books. The RROS IC automatically translates all signals so they can be used through the rBeacon() function. We even offer preprogrammed beacon chips for building your own beacons.
- A servo-turret mounted ranging sensor is fully supported when digital perimeter sensing is utilized. It is NOT needed though when VIRTUAL sensors are created from ranging sensors. When ranging sensors are used, the system automatically extracts both rRange() information and proximity information from the perimeter sensors themselves.
- One electronic compass, the HMC6352 compass module previously available from Parallax, and still available from the RobotShop, Sparkfun, and JameCo, is currently supported.
- Many expansion features are present in the RROS. We think advanced hobbyists will find it very versatile. It will, for example, allow you to connect one or more microcontrollers (such as the Parallax Stamp, Arduino processors, or the Pololu Orangutan controllers) to a special serial bus. These controllers can provide camera or GPS data or handle any custom sensory system you wish. The RROS

documentation provides all the details you need to handle normal operations or to customize your system to meet advanced needs. We encourage companies that distribute advanced sensors to offer RROS compatible versions and we will certainly work with vendors to ensure that RobotBASIC users will have access to a wide variety of options.

- The RROS can even be used with steerable robots such as those available from Minds-i (some restrictions apply). Four wheel robots are very stable in outdoor environments and have advantages over 2-wheel robots in some applications. (Two-wheeled bots like our simulator are much better for the home/office environment).
- The RROS chip has a dual personality. When two RROS chips are used together, one of the chips can turn itself into an Arm Expansion that allows RobotBASIC to control 5 servomotors (for shoulder, elbow, wrist, wrist rotate, and hand open/close) over the standard communication link. The Arm Expansion also provides the ability to acquire additional analog and digital data so that your arm can have appropriate sensory information (example: pressure at the gripper's fingertips, determining if something is in the hand, etc).

About the author

John Blankenship spent decades teaching electronics, programming, and robotics and the college level. His love of teaching, and a desire to help students, inspired him to co-develop the RobotBASIC programming language with Samuel Mishal. You can use this link [†] to view an interview with John, published by Circuit Cellar Magazine:

Email: RobotBASIC ≫at≪ yahoo.com

[†] http://circuitcellar.com/community/interviews/a-love-of-teaching-a-lifetime-of-robotics-an-interview-with-john-blankenship/